DRILL-ACROSS QUERIES IN TEMPORAL SCHEMES

Andrés F. Jiménez B., Francisco J. Moreno A.

Universidad Nacional de Colombia, Sede Medellín.
Facultad de Minas, Escuela de Ingeniería de Sistemas e Informática
Tel.: (094) 4255376, Fax: (094) 4255365
E-mail: {afjimene, fjmoreno}@unal.edu.co

Abstract: Data warehouses systems use multidimensional schemes to integrate a lot of information that allows the analysis of the behaviour of specific areas in the organizations. In practice, the changes in dimensions are frequent. These changes yield losses of valuable information. Also it is necessary to cross information that can be found in some multidimensional schemes, this operation is called drill-across. In this article we propose a mechanism in order to cross information between two multidimensional schemes, with support for changes in dimension instances, i.e, temporal multidimensional schemes. Moreover, we define the necessary conditions for crossing the schemes and show the respective query language

Keywords: OLAP; Drill-Across; TOLAP; Temporal Schemas; Data Warehouses; Multidimensional Models.

1. INTRODUCTION

OLAP (On-Line Analytical Processing) (Kimball, 1998) tools allow the organizations to generate summary reports of information about behavior of areas, for example, sales, payments, shippings, among others. The information system use for OLAP tools is a Multidimensional Schemes (MS).

A MS is generally based on a fact table that marks particular information for a date, for example a sale, an incoming inventory, a financial transaction, etc. Using this information is possible to generate consolidated following the levels in each dimension.

These MSs could be related (have dimensions in common) with others MSs. In this case, it is feasible to consolidate information navigating between them. This specific operation is called drill-across in OLAP (Golfarelli *et al*., 1998).

The body of the article is: Section 2 shows how to describe a temporal MS, Section 3 describes the process of crossing information between different MSs (drill-across), Section 4 describes TOLAP language, Section 5 defines and shows the drill-across process with temporal MSs. Finally, Section 6 presents the conclusion and future works.

2. TEMPORAL MULTIDIMENSIONAL SCHEMES

MSs show a problem for updates in the instances and the structure of the dimensions when the business requirements change. This problem is recognized in (Kimball, 1998) and it is called *slowly changing dimensions.*

Let us suppose we have an inventory system where you keep the monthly information of product’s inventory, at the same time, this inventory will be associated with a manufacture's center. A manufacture's center is related (assigned) with a specific city. What happens if this manufacture's center changes of city? If you do not have the history of changes, then you lose the valuable information.

Figure 1 shows a MS inventory. It shows how to group the information around the different dimensions. It must be noted that each dimension is composed by levels, for example, the dimension Time is composed by levels month and year.

There is a temporal relation (T) between the manufacture's center and the cities, this means that a manufacture's center can be related to city X from 1986 to 1991 and change in 1991 to another city Y and remains there until now.

A sample of the fact table of this MS inventory can be seen in Table 1.

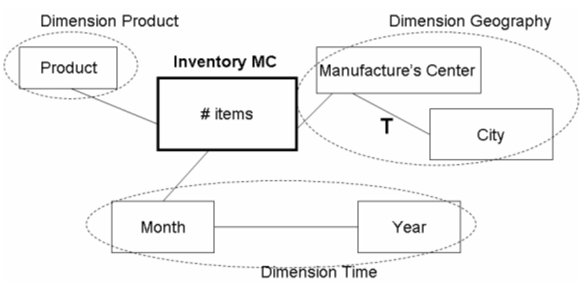

Fig. 1. Multidimensional scheme information of Inventory Manufacture's Center

Table 1. Product inventory in manufacture's center

Product	Manufacture's Center	# items	Time
p1	mc1	200	m1
p2	Mc1	600	m2
p1	mc2	300	m1
p2	mc2	400	m2
p1	mc1	300	m3
p2	mc1	500	m4
p1	mc2	300	m3

In Table 1 can be appreciated that the only information associated to the dimension Geography is the manufacture's center, if we need the respective city to which an inventory registry belongs it is necessary to navigate through the relation of manufacture's center to the specific city. In the case in which a manufacture's center changes of city in a specific time, *the fact table will not have knowledge of that event.* Just as it is presented in (Snodgrass, 1995) it is required to manage the registration of these changes.

Table 2 shows a way to represent the information for a temporal relationship between a city and a manufacture's center.

Table 2. Management of temporal relationship between the levels from a dimension

loLevel	upLevel	loVal	upVal	From	To
MCenter	City	mc1	c22	t1	t4
MCenter	City	mc1	c23	t4	Now
MCenter	City	mc2	c30	t1	Now

Table 2 shows how to register the information for a period of time in which the manufacture's center belongs to a city. The keyword *now* (Vaisman and Mendelzon, 2001) establishes the present date.

3. DRILL-ACROSS OPERATION

Drill-across operation combines two MSs in such a way that can be queried like a single unified MS.

It is shown (Golfarelli et al, 1998) it is possible to compare measures generated from different related MSs using a series of basic rules that allow generate a new MS from the MSs used in the operation.

The rules are:

a) The measures of the original MSs are added to the new one.
b) In the new MS only remains shared levels between the two MSs. They must share the same semantic meaning too.
c) Only remains the relationships of the share levels.
d) Only remains the process of aggregation that can be applied on the measures using the shared levels.

Figure 2 shows an inventory MS, it stores inventory information of a store.

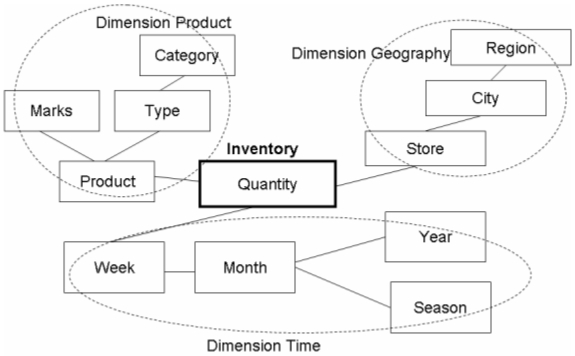

Fig. 2. Inventory scheme

Let us suppose, we have another MS. It is oriented to the shipment information of the products toward different clients. This MS is presented in Figure 3.

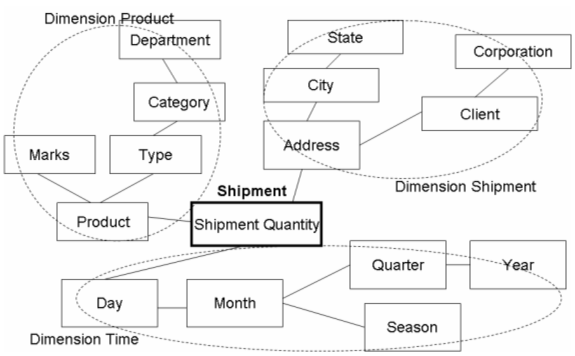

Fig. 3. Shipment scheme

Both MSs share some levels that can define a new MS that cross information from shipment versus inventory products. The resultant MS is generated after applying the rules shown at the beginning of this section. The MS is presented in Figure 4.

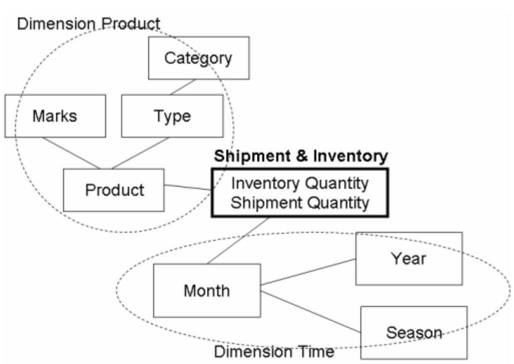

Fig. 4. Resultant MS inventory and shipment

In the resultant MS in Figure 4 it is possible to recognize that it preserves the dimension product and the dimension time.

The levels city and state although they appear in the original MSs they do not appear in the new MS because they do not share the same semantic meaning. City represents the store location in one of them and in the other represents the destiny of the shipments.

4. TOLAP

One of the first proposed languages to deal with temporal properties was TSQL2 (Snodgrass, 1995). Although this language can be used to query a temporal relational model, TSQL2 is not suitable to query MSs. SQL/TP (Toman, 1997), and HSQL (Sarda Morris, 1993) are other languages for querying temporal information, but neither have the properties necessary for temporal MSs.

On the other hand TOLAP (Vaisman and Mendelzon, 2001) is a language that allows the conception of queries about temporal MS. This language is appropriate for extracting information keeping in mind the temporal changes in instances and structures.

In order to demonstrate the expressive power of TOLAP, we will use the example shown in (Mendelzon and Vaisman, 2000) about a basket tournament. A simple question like "How many points did a Blazer's player score?" this sentence has two interpretations: The first consists in adding all the points scored ever by the player (he belongs currently to the blazers) including the points scored in another team. The second interpretation consists in adding only those points scored by the player with Blazer team.

For the first interpretation the following query can be posed in TOLAP:

$$Q(x, SUM(p)) \leftarrow Points(x,p,t),$$
$$x\ [Now] \rightarrow team:'Blazers'.$$

The fact table is Points and from it three parameters are extracted: The player denoted by **x**, the points denoted by **p**, and the time denoted by **t**. In the second line it is shown how to group the information using the relationship that exists between the player and the team which belongs utilizing the operation Rollup in terms of OLAP (in this case it is applied for the instance Blazers). The keyword *now* means that only is verified the ownership of the player to the team Blazers *currently* but it will add all the points scored by the player although they were scored in another team.

The second interpretation is solved by means of the following query:

$$Q(x, SUM(p)) \leftarrow Points(x,p,t),$$
$$x\ [t] \rightarrow team{:}'Blazers'.$$

In the second query the operator that describes the ownership to the level team uses the modifier t; this means that it will add only the points scored by the players when he belonged (or belongs) to the Blazer team. This is independent from the current ownership of the player to the Blazer team, this means that it is possible to add points of players playing in another team.

Although TOLAP was conceived for queries in MSs, it does not have explicitly operators for drill-across in order to retrieve information from more than one MS.

5. DRILL-ACROSS IN TEMPORAL MS

In Figure 5 a temporal MS related with the one shown in Figure 1 is shown.

In the MS of Figure 5 exists a temporal relationship between the levels in the dimension Product, this means that a product can change the category through time. A query that may require these two MSs is: "to get the number of monthly inventory of the products for a specific city and a specific category."

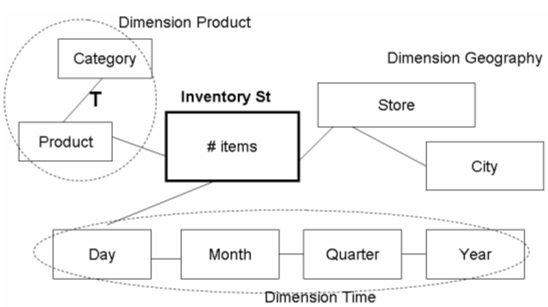

Fig. 5. Store inventory scheme

To facilitate this process, it is possible to use a drill-across operation in such a way that allows to obtain different measures extracted from two related MSs; note that this query does not keep in mind a semantic recognition of the data (Abelló *et al*, 2002), this means, in the case the MSs share common dimensions and levels these should have the same names. See Section 5 for more details about this.

Once it is recognized that there are dimensions shared between two MSs it is possible to determine the structure for the resultant dimensions, in the following way extending the proposal of (Golfarelli, *et al*, 1998):

1. The identical levels are conserved (that are shared in the MSs) although these have been born in different levels.
2. In the case a MS has a level Z that groups another level that is common between the two MSs but Z is not found in the other MS, it will be also able to be conserved in the resultant MS. In the proposal of (Golfarelli, *et al*., 1998) this was not carried out since only the shared levels are conserved.
3. The measures in the fact table are added in the final fact table.

Under these restrictions the drill-across operation between the two MSs shown previously can be observed in the Figure 6, where are shown the levels which remain after applying the operation.

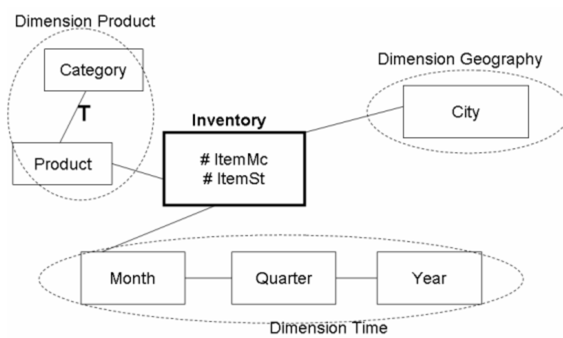

Fig. 6. Resultant scheme for drill-across operation

In the Table 3 an example of the contained registers in the fact table is presented in resultant of the drill-across operation between the two EMs.

<u>Table 3. Resultant fact table in the drill-across operation</u>

Product	City	Time	itemMC	ItemSt
p1	C22	m1	200	0
p2	C22	m2	600	0
p1	C30	m1	300	0
p2	C30	m2	400	0
p1	C22	m3	300	0
p2	C23	m4	500	0
p1	C30	m3	300	0
p1	C22	m1	0	200
p2	C22	m2	0	600

p1	C22	m1	0	300
p2	C22	m2	0	400
p1	C22	m1	0	300
p2	C22	m2	0	500
p1	C22	m1	0	300

Our proposal is to extend TOLAP with a drill-across operation expressed thus:

Inventory(M,Sum(IT)+Sum(ICP))
←InventoryMC DRILL ACROSS InventoryST
(Product:P,Month:M,City:C,itemSt:IT,itemMc:IMC),
P[M] → Category:'Cat1', C.CityID = C22

This query simplifies a lot the way to query the information since with a few lines of code it is possible to generate a consolidated of monthly information of the inventory of products of a specific city (C22) and a specific category (Cat1) . The same query in SQL can be defined in the following way:

```
SELECT month, SUM( itemMc + itemSt )
FROM
(SELECT T.month, SUM(I.itemMc) AS itemMc, 0 AS itemSt
FROM inventoryMc I, Product P, MCenter MC, Time T
WHERE I.item = P.loVal AND P.loLevel = 'product'
  AND P.upLevel = 'category' AND P.upVal = 'Lacteos'
  AND I.MCenter = CP.loVal AND CP.loLevel = 'MCenter'
  AND MC.upLevel = 'city' AND T.day = I.day
  AND P.From <= I.day AND I.day <= P.To
  AND MC.From <= I.day AND I.day <= MC.To
  AND MC.upVal = 'C22'
GROUP BY P.upVal, T.month
UNION
SELECT T.month, 0 AS itemMc, SUM(I.itemSt) AS itemSt
FROM inventorySt I, Product P, Store ST, Time T
WHERE I.item = P.loVal AND P.loLevel = 'product'
  AND P.upLevel = 'category' AND I.store = ST.idStore
  AND ST.city = 'C22' AND T.day = I.day
  AND P.From <= I.day AND I.day <= P.To
GROUP BY P.upVal, T.month)
GROUP BY product, month
```

In SQL it is required to specify how the different levels of the dimensions are related and how they apply the restrictions in the different fact tables.

6. CONCLUSIONS

The queries in TOLAP are more simple, elegant and intuitive than their corresponding versions in SQL. By means of TOLAP a clear use of the dimensions is done and allows additionally to handle the temporal relationships that are often presented in the instances of the dimensions.

The process of crossing MSs allows for extracting information directly in a single query from the resultant MS, which includes the levels shared among the original MSs.

Once it is possible to cross information among the different MSs involved in the query it is necessary to improve the aspect of the performance of the same one, since the query requires the processing of millions of data that are found in the fact tables.

An aspect that we do not treat is when levels have different names but have the same meaning. We think it is necessary to develop a model that permits to cross dimensions with equal semantics but different nomenclatures, ontologies can help in this task.

REFERENCES

Abelló A., Samos. J., and Saltor. F. (2002). On Relationships Offering New Drill-across Possibilities (© ACM). In 5th International Workshop on Data Warehousing and OLAP (DOLAP 2002). McLean (USA), November 2002. Pages 7-13. ACM Press, ISBN 1-58113-590-4.

Golfarelli M., Maio D. y Rizzi. S. (1998). The Dimensional Fact Model: A conceptual Model for Data Warehouses. Int. Journal of cooperative Information systems. Bologna College. Italy.

Kimball. R. (1998). The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses. Ed. Wiley. Second Edition.

Mendelzon. A. and Vaisman. A. (2000). Temporal queries in OLAP. In Proceedings of the 26th VLDB Conference, Cairo, Egypt.

Sarda. N.L. (1993). HSQL: A Historical Query Language Temporal Databases. Theory, Design and Implementation. Benjamin/Cummings.

Snodgrass. R. (1995). The TSQL2 Temporal Query Language. Kluber Academic Publishers. Fundamental Concepts.

Toman. D. (1997). A Point-based Temporal Extension to sql. In Proceedings of DOOD'97, Mountreaux, Switzerland.

Vaisman A. y Mendelzon A. (2001). A Temporal Query Language for OLAP: Implementation and a Case Study. DBPL 2001. Rome, Italy.